# Weekly Project Report (COMP-10199)

| | |
|---|---|
| Project Title | Project 14: Voice Agent for Candidate Onboarding & Career Guidance |
| Report Date | 2026/02/06 |

| Team Members (Full name) | |
|---|---|
| 1 – Dhairya Patel | 2 – Brendan Dileo |
| 3 – Jay Choksi | 4 – Rishyu Babariya |

**What was accomplished this week? (Please include details and avoid general terms)**

- Reviewed and Updated Project Proposal after receiving feedback from the Project Owner.
- Updated the UI Dashboard Mockup based on Project Owner feedback.
- Began implementing basic dashboard code based on updated mockup designs.
- Created the core technical structure of the repository, prioritizing focused files and classes.
- Organized related files and classes into folders/directories and Python modules for maintainability.
- Implemented core files related to Prototype Slice 1 (Voice Agent Functionality).
- Developed Voice Agent using OpenAI for TTS, LLM, and STT integration.
- Met internally as a team to discuss individual expectations, progress, and project status.
- Tested cloning and executing first prototype application across team members laptops to ensure portability.
- Encountered OpenAI service outage, prompting development of local model implementation as backup.
- Implemented fully local voice agent prototype for system redundancy and client “plug and play” requirements.
- Identified Python version compatibility and dependency issues during cross-machine testing.
- Updated project documentation with setup instructions and testing procedures.
- Submitted two pull requests to the main branch following code review process.

**What was not completed as planned? Please explain why and state your plan to fix it.**

- Full cross-team testing and deployment readiness were not completed this week.
  - Reason: Python version compatibility and dependency conflicts discovered during portability testing on team members laptops. Different system configurations and Python versions caused installation failures.
  - Plan to Fix: Document specific Python version requirements, update requirements.txt with exact dependency versions, accepted range of versions, and document troubleshooting steps for common setup issues. Will test the updated configuration across all team member machines next week to ensure consistent environment setup before proceeding with further development.

## Team Member 1 Activities (Please include details and avoid general terms)

- Performed detailed code reviews for pull requests submitted by team members for Prototype Slice 1 prior to merging into the main branch.
- Validated functionality, code structure, and integration readiness across contributions from multiple team members as part of the testing role.
- Conducted cross-machine setup and execution testing by assisting with environment configuration on team members' laptops.
- Identified Python version and dependency compatibility issues, particularly related to Streamlit and environment alignment.
- Assisted in troubleshooting and documenting dependency issues to support consistent setup across different systems.
- Updated and refined the project proposal document based on feedback and requirements requested by the client.
- Resolved repository access issues by coordinating with the repository handler to grant appropriate permissions for team members.
- Tested both the local voice agent implementation and the OpenAI-based voice agent to verify functionality and system behavior during service outages.

## Team Member 2 Activities (Please include details and avoid general terms)

- Branched off main in GitHub repository to develop Prototype Slice 1 voice agent logic.
- Created the project's technical structure (src/, app/, logs/, tests/ directories).
- Implemented core voice agent functionality using OpenAI's API (TTS, LLM, STT).
- Updated documentation with setup and testing instructions for team execution.
- Revised requirements file and virtual environment setup for environment replication.
- Pushed feature branch to remote repository and opened pull request to main requiring code review.
- Addressed OpenAI service outage by experimenting with local models for clients "plug and play" requirement.
- Implemented a fully local voice agent prototype for system redundancy and to address group OpenAI account needs.
- Pushed local voice agent source code to remote repository and submitted pull request for merge to main.
- Tested prototype portability by cloning and executing code on team members laptops.
- Identified Python version compatibility and dependency issues during cross-machine testing.
- Began troubleshooting version and dependency issues for resolution in the following week.

## Team Member 3 Activities (Please include details and avoid general terms)

- Designed and implemented RESTful APIs to support communication between the frontend and backend components of the system.
- Utilized Python as the primary language and incorporated session management to maintain user interaction state across requests.
- Documented the API endpoints and usage details to support future integration and team understanding.
- Defined a structured logging strategy, including folder organization for sessions, metrics, runtime errors, and system events.
- Planned log capture for key runtime events such as system failures, timestamps, session identifiers, latency, and user confirmation behavior.
- Established guidelines to prevent committing real log data to the public repository while ensuring directory tracking using .gitkeep.
- Began outlining a centralized logs_manager.py module to manage and standardize logging functionality in future integration phases.
- Considered logging best practices and data safety by planning to exclude real runtime logs from the public repository while maintaining directory structure for future integration.

### Team Member 4 Activities (Please write down details and avoid general terms)

- Designed and implemented the initial Streamlit-based dashboard interface for the AI Voice Assistant prototype.
- Created a user-friendly layout to capture voice input and display the assistant’s responses in a readable paragraph format.
- Implemented structured form fields within the dashboard to extract and display onboarding information, beginning with the user’s name.
- Aligned frontend implementation with updated dashboard mockups and feedback provided by the Project Owner.
- Established the foundational UI required to integrate the voice agent pipeline with the candidate onboarding workflow.
- Focused on ensuring clarity, usability, and readiness for future backend and voice agent integration.
- Ensured the dashboard layout was structured to support future backend and voice agent integration without requiring major UI refactoring.

### Did you have meeting with your Project Owner? If yes, write details of meeting and its outcome

For this week we did not have a meeting with client.

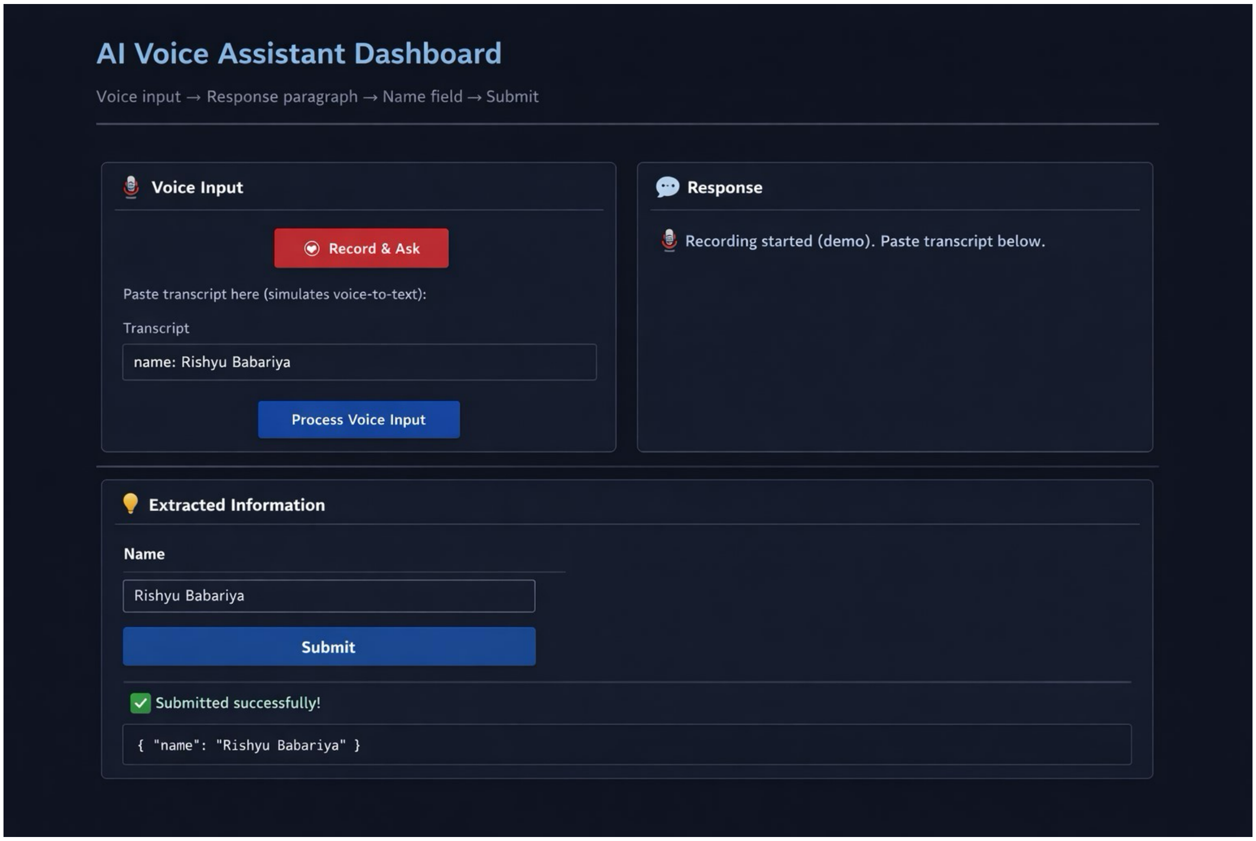
### For major GitHub commits during the week, specify the Commit Title and its reason

- Setup Technical Project Structure (2026/02/02)
  - Established foundational project architecture (src/, app/ directories) for voice agent development.
- Implement Core Voice Agent Logic and Update Requirements (2026/02/03)
  - Integrated OpenAI API functionality and updated dependencies for team environment replication.
- Setup Basic Context Management and Field Prompt (2026/02/03)
  - Implemented conversation context handling and initial prompt configuration for voice interaction flow.
- Merge Pull Request #1 from feat/voice-agent (2026/02/04)
  - Merged initial voice agent prototype source code to main branch following code review.
- Create and Implement LocalVoiceAgent implementation (2026/02/05)
  - Developed a local model alternative in response to OpenAI outage and client “plug and play” requirements.
- Merge Pull Request #2 - LocalVoiceAgent Implementation (2026/02/05)
  - Merged local voice agent feature to main branch to provide system redundancy option.

### Is there any other information you want to reflect about your project (optional)

- Mitigated potential development risks by developing dual voice agent options (OpenAI + Local).
- Discovered the importance of early cross-machine testing to catch environmental issues sooner.

If you have any screenshot, artifact, etc. that can illustrate your week's progress, please include it here (Optional)



AI Voice Assistant Dashboard

Voice input → Response paragraph → Name field → Submit

Voice Input

Paste transcript here (simulates voice-to-text):

Transcript

name: Rishyu Babariya

Response

Recording started (demo). Paste transcript below.

Extracted Information

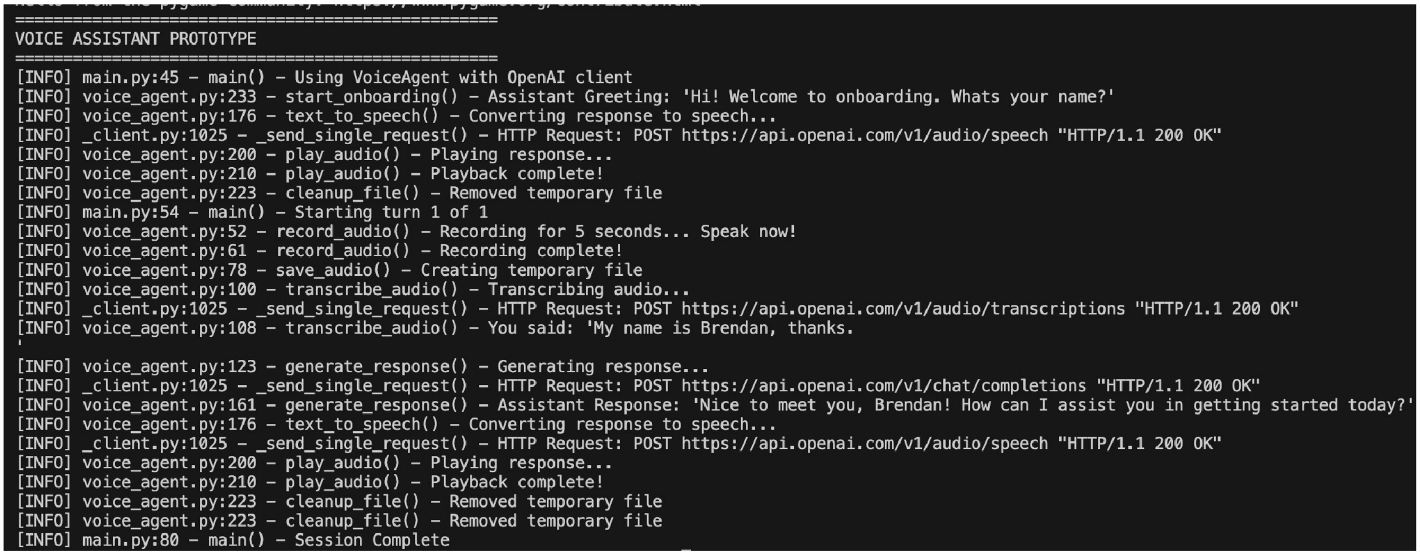
Name

Rishyu Babariya

✓ Submitted successfully!

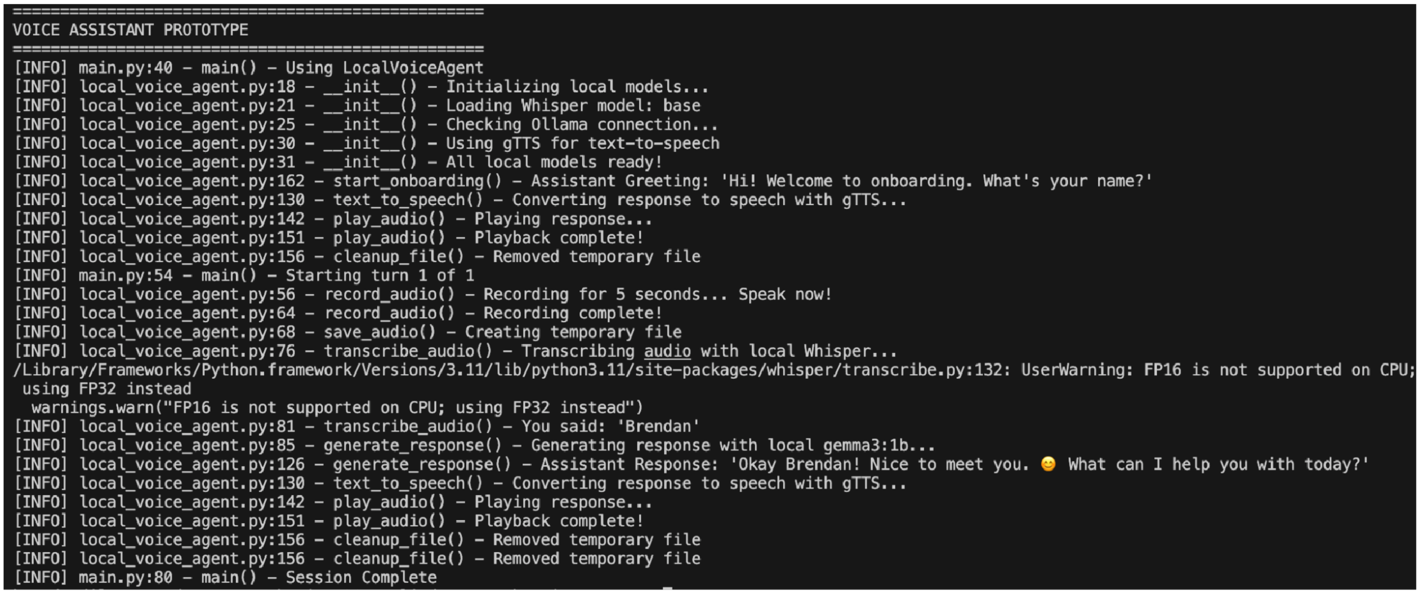
```
{ "name": "Rishyu Babariya" }
```

Figure 1: Updated Dashboard Mockup



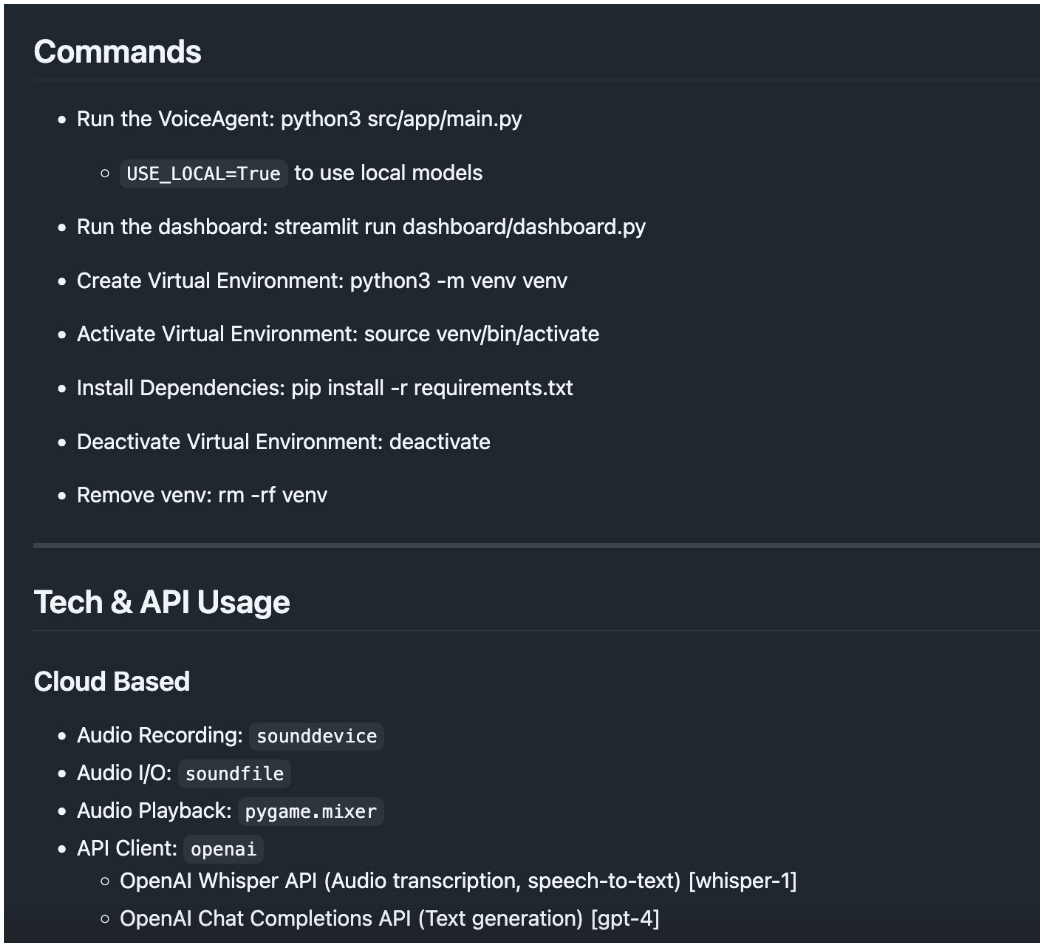
```
Hello from the pygame community. https://www.pygame.org/contribute.html
==================================================
VOICE ASSISTANT PROTOTYPE
==================================================
[INFO] main.py:45 - main() - Using VoiceAgent with OpenAI client
[INFO] voice_agent.py:233 - start_onboarding() - Assistant Greeting: 'Hi! Welcome to onboarding. Whats your name?'
[INFO] voice_agent.py:176 - text_to_speech() - Converting response to speech...
[INFO] _client.py:1025 - _send_single_request() - HTTP Request: POST https://api.openai.com/v1/audio/speech "HTTP/1.1 200 OK"
[INFO] voice_agent.py:200 - play_audio() - Playing response...
[INFO] voice_agent.py:210 - play_audio() - Playback complete!
[INFO] voice_agent.py:223 - cleanup_file() - Removed temporary file
[INFO] main.py:54 - main() - Starting turn 1 of 1
[INFO] voice_agent.py:52 - record_audio() - Recording for 5 seconds... Speak now!
[INFO] voice_agent.py:61 - record_audio() - Recording complete!
[INFO] voice_agent.py:78 - save_audio() - Creating temporary file
[INFO] voice_agent.py:100 - transcribe_audio() - Transcribing audio...
[INFO] _client.py:1025 - _send_single_request() - HTTP Request: POST https://api.openai.com/v1/audio/transcriptions "HTTP/1.1 200 OK"
[INFO] voice_agent.py:108 - transcribe_audio() - You said: 'My name is Brendan, thanks.
'
[INFO] voice_agent.py:123 - generate_response() - Generating response...
[INFO] _client.py:1025 - _send_single_request() - HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
[INFO] voice_agent.py:161 - generate_response() - Assistant Response: 'Nice to meet you, Brendan! How can I assist you in getting started today?'
[INFO] voice_agent.py:176 - text_to_speech() - Converting response to speech...
[INFO] _client.py:1025 - _send_single_request() - HTTP Request: POST https://api.openai.com/v1/audio/speech "HTTP/1.1 200 OK"
[INFO] voice_agent.py:200 - play_audio() - Playing response...
[INFO] voice_agent.py:210 - play_audio() - Playback complete!
[INFO] voice_agent.py:223 - cleanup_file() - Removed temporary file
[INFO] voice_agent.py:223 - cleanup_file() - Removed temporary file
[INFO] main.py:80 - main() - Session Complete
```

Figure 2: OpenAI API Voice Agent Flow



```
==================================================
VOICE ASSISTANT PROTOTYPE
==================================================
[INFO] main.py:40 - main() - Using LocalVoiceAgent
[INFO] local_voice_agent.py:18 - __init__() - Initializing local models...
[INFO] local_voice_agent.py:21 - __init__() - Loading Whisper model: base
[INFO] local_voice_agent.py:25 - __init__() - Checking Ollama connection...
[INFO] local_voice_agent.py:30 - __init__() - Using gTTS for text-to-speech
[INFO] local_voice_agent.py:31 - __init__() - All local models ready!
[INFO] local_voice_agent.py:162 - start_onboarding() - Assistant Greeting: 'Hi! Welcome to onboarding. What's your name?'
[INFO] local_voice_agent.py:130 - text_to_speech() - Converting response to speech with gTTS...
[INFO] local_voice_agent.py:142 - play_audio() - Playing response...
[INFO] local_voice_agent.py:151 - play_audio() - Playback complete!
[INFO] local_voice_agent.py:156 - cleanup_file() - Removed temporary file
[INFO] main.py:54 - main() - Starting turn 1 of 1
[INFO] local_voice_agent.py:56 - record_audio() - Recording for 5 seconds... Speak now!
[INFO] local_voice_agent.py:64 - record_audio() - Recording complete!
[INFO] local_voice_agent.py:68 - save_audio() - Creating temporary file
[INFO] local_voice_agent.py:76 - transcribe_audio() - Transcribing audio with local Whisper...
/Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages/whisper/transcribe.py:132: UserWarning: FP16 is not supported on CPU;
 using FP32 instead
  warnings.warn("FP16 is not supported on CPU; using FP32 instead")
[INFO] local_voice_agent.py:81 - transcribe_audio() - You said: 'Brendan'
[INFO] local_voice_agent.py:85 - generate_response() - Generating response with local gemma3:1b...
[INFO] local_voice_agent.py:126 - generate_response() - Assistant Response: 'Okay Brendan! Nice to meet you. 😊 What can I help you with today?'
[INFO] local_voice_agent.py:130 - text_to_speech() - Converting response to speech with gTTS...
[INFO] local_voice_agent.py:142 - play_audio() - Playing response...
[INFO] local_voice_agent.py:151 - play_audio() - Playback complete!
[INFO] local_voice_agent.py:156 - cleanup_file() - Removed temporary file
[INFO] local_voice_agent.py:156 - cleanup_file() - Removed temporary file
[INFO] main.py:80 - main() - Session Complete
```

Figure 3: Local Voice Agent Execution Flow



## Commands

- Run the VoiceAgent: python3 src/app/main.py
  - `USE_LOCAL=True` to use local models
- Run the dashboard: streamlit run dashboard/dashboard.py
- Create Virtual Environment: python3 -m venv venv
- Activate Virtual Environment: source venv/bin/activate
- Install Dependencies: pip install -r requirements.txt
- Deactivate Virtual Environment: deactivate
- Remove venv: rm -rf venv

## Tech & API Usage

### Cloud Based

- Audio Recording: `sounddevice`
- Audio I/O: `soundfile`
- Audio Playback: `pygame.mixer`
- API Client: `openai`
  - OpenAI Whisper API (Audio transcription, speech-to-text) [whisper-1]
  - OpenAI Chat Completions API (Text generation) [gpt-4]

Figure 4: Documented commands and tech usage